Reproducible workflows with Jupyter: case study in materials simulation

Hans Fangohr[1,2,9], Martin Lang[1,9], Samuel J. R. Holt[1,9], Swapneel Pathak[1,9], Min Ragan-Kelley[3], Marijan Beg[4,2], Juliette Belin[5], Thomas Kluyver[6], Olexandr Konovalov[7], Nicolas M. Thiéry[8]

1) Max Planck Institute for the Structure and Dynamics of Matter, Hamburg, Germany; 2) University of Southampton, Southampton, United Kingdom; 3) Simula Research Laboratory, Oslo, Norway; 4) Imperial College, London, United Kingdom; 5) Logilab, Paris, France; 6) European XFEL, Schenefeld, Germany; 7) University of St Andrews, St Andrews, United Kingdom; 8) Université Paris-Saclay, Orsay, France; 9) Center for Free-Electron Laser Sciene (CFEL)

Key Words: *Reproducibility, Workflows, Jupyter, Python, Packaging, Binder, Computing Methods.*

Abstract

Using “Ubermag” as a case study in materials research, we describe how tools of Project Jupyter can be used to support more reproducible research.

Ubermag is based on a Python library [1] that acts as an interface to two existing research simulation tools (OOMMF and mumax3, which are based on Tcl/C++ and Go/CUDA, respectively) and which have comparable simulation functionality. Through the Python interface, simulations can be conducted from Jupyter notebooks. The whole project, bringing together the Python interface with additional analysis tools, is known as Ubermag [2], and available as open source [3]. Data analysis is also conducted from the Jupyter notebook. For many problems, an iterative and explorative cycle is possible and effective. The Ubermag software allows to simulate magnetic materials at the micrometre scale, but many of the concepts should be transferable to other domains.

We discuss how the setup can help scientists to move to more reproducible workflows and publications [4]. This includes the one-study one-document nature of the approach to capture the steps that have been carried out. For more reproducible publications, we propose to publish a set of Jupyter notebooks with each publication, where the notebooks are used to compute central figures and statements of the paper [4].

A challenge is the preservation and creation of software environments in which the notebooks can be executed (potentially using software outside the notebook, which might be called from the notebook). We mention Binder [5] as a possible option here.

We discuss other benefits of the notebook-based approach to computational science, including no-install creation of software environments via Binder, easy documentation of software using notebooks as Sphinx input, executable tutorials (with Binder) and automatic testing of documentation and reproducibility using nbval [6].

References

[1] Marijan Beg, Ryan A. Pepper, Hans Fangohr, "User interfaces for computational science: a domain specific language for OOMMF embedded in Python", AIP Advances 7, 056025 (2017) https://doi.org/10.1063/1.4977225

[2] Hans Fangohr, Martin Lang, Samuel J. R. Holt, Swapneel Amit Pathak, Kauser Zulfiqar, Marijan Beg “Vision for unified micromagnetic modeling (UMM) with Ubermag”, AIP Advances **14**, 015138 (2024) https://doi.org/10.1063/9.0000661

[3] Ubermag home page https://ubermag.github.io and sources (2022) https://github.com/ubermag

[4] Marijan Beg, Juliette Belin, Thomas Kluyver, Alexander Konovalov, Min Ragan-Kelley, Nicolas Thiery, Hans Fangohr, "Using Jupyter for reproducible scientific workflows", Computing in Science & Engineering 23, 36-46 (2021) https://doi.org/10.1109/MCSE.2021.3052101

[5] Project Jupyter, M. Bussonnier, J. Forde, J. Freeman, B. Granger, T. Head, C. Holdgraf, K. Kelley, G. Nalvarte, A. Osheroff, M. Pacer, Y. Panda, F. Perez, B. Ragan Kelley, and C. Willing, “Binder 2.0 - Reproducible, interactive, shareable environments for science at scale”, Proceedings of the 17th Python in Science Conference, pp. 113 – 120 (2018) https://jupyter.org/binder

[6] Hans Fangohr, Vidar Fauske, Thomas Kluyver, Maximilian Albert, Oliver Laslett, David Cortes-Ortuno, Marijan Beg, Min Ragan-Kelly. "Testing with Jupyter notebooks: NoteBook VALidation (nbval) plug-in for pytest" https://arxiv.org/abs/2001.04808 (2020) https://github.com/computationalmodelling/nbval